21
An Improved GA Based Modified Dynamic Neural Network for Cantonese-Digit Speech Recognition

[1]S.H. Ling, [2]F.H.F. Leung, [2]K.F. Leung, [3]H.K. Lam, and [1]H.H.C. Iu
[1] *The University of Western Australia, WA, Australia*
[2]*The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong*
[3] *Division of Engineering, King's College London, Strand, London, United Kingdom*

1. Introduction

An artificial Neural Network (ANN) is a well known universal approximator to model smooth and continuous functions (Brown & Harris, 1994). As ANNs can realize nonlinear models, they are flexible in modeling a wide variety of real-world complex applications, such as handwriting recognition, speech recognition, fault detection, medical inspection (Zhang, 2000), etc. ANNs being applied for pattern classification can be divided into two main categories: static and dynamic. Static pattern classification problems are usually tackled by multi-layer perceptron (MLP), radial basis feed-forward (RBF) networks and learning vector quantization (LVQ). However, limited by its structure of a traditional, a feed-forward network cannot model the correlation between the previous time frames and the current time frame. Thus, some dynamic applications, such as speech recognition, time varying prediction, dynamic control, etc. are difficult to be realized by static neural networks. Neither can feed-forward neural networks deal with problems without a fix dimension of input patterns. Recurrent Neural Network (RNN) (Engelbrecht, 2002; Kirschning et al., 1996; Zhang et al., 1993) and Time Delay Neural Network (TDNN) (Waibel et al., 1989) are used to overcome the limitations of feed-forward networks. They dynamically model time series cases; in other words, they are predictor networks that predict the next data frame from the current data frame.

ANNs operate in two stages: learning and generalization. Learning of a neural network is to approximate the behavior of the training data while generalization is the ability to predict well beyond the training data (Zhang, 2000). In order to have a good learning and generalization ability, a good tuning algorithm is needed. In this chapter, Genetic Algorithm (GA) is used as the tuning algorithm for training neural networks.

GA is a directed random search technique (Hanaki et al., 1999; Michalewicz, 1994; Pham & Karaboga, 2000) that is widely applied in optimization problems (Hanaki et al, 1999; Michalewicz, 1994; Pham & Karaboga, 2000). It is especially useful for complex optimization problems when the number of parameters is large and the analytical solutions are difficult to obtain. GA can help find out the globally optimal solution over a domain. It has been applied in different areas such as fuzzy control (Leung et al., 2004), path planning, modeling and classification (Setnes & Roubos, 2000), tuning parameters of neural/neural-fuzzy networks (Leung et al., 2003; Ling et al., 2003) etc. A lot of research efforts have been spent to improve the performance of GA. Different selection schemes and genetic operators

Source: Robust Speech Recognition and Understanding, Book edited by: Michael Grimm and Kristian Kroschel,
ISBN 987-3-90213-08-0, pp.460, I-Tech, Vienna, Austria, June 2007

have been proposed. Selection schemes such as rank-based selection, elitist strategies, steady-state election and tournament selection have been reported (Davis, 1991). There are two kinds of genetic operations, namely crossover and mutation. Apart from random mutation and crossover, other crossover and mutation mechanisms have been proposed. As the crossover mechanisms, two-point crossover, multipoint crossover, arithmetic crossover and heuristic crossover have been reported (Davis, 1991; Michalewicz, 1994; Srinivas & Patnaik, 1994). As the mutation mechanisms, boundary mutation, uniform mutation and non-uniform mutation can be found (Davis, 1991; Michalewicz, 1994; Srinivas & Patnaik, 1994).

In this chapter, a dynamic neural network tuned by an improved GA (Lam et al., 2004) is proposed. New genetic operations (crossover and mutation) will be introduced. Rules have been introduced to the crossover process to make offspring widely spread along the domain. A fast convergence rate can be reached. A different process of mutation has been applied. The proposed dynamic neural network of architecture shown in Fig. 1 consists of two modules: a tuner neural network (TNN) and a classifier recurrent neural network (CRNN). This specific architecture provides a one-input-one-rule property to the network. By using the TNN, some parameters can be determined from the input patterns and applied to the CRNN for further classification. In general, a traditional RNN can only have one set of fix parameters to model all different input patterns and their time variations owing to the limitation of its structure. In practice, consider the case depicted by Fig. 2; the data sets S1 & S2 belong to the class 1 but they are separated far apart, and the data set S3 belong to the class 2 in the spatial domain. When a traditional RNN with an inadequate number of parameters is used, the network will only be trained to recognize a data set R between S1 and S2. Then, S3 could be misclassified as class 1 as shown in Fig. 2(b). The recognition accuracy will then be lowered. However, if the number of parameters is large, the number of iteration required in the training process will be increased. In order to reduce the number of parameters of the network, we propose the dynamic NN. On using this network, when the input data belongs to S1, the TNN will provide the parameter set 1 for the CRNN to handle the data set S1. When the input data S2 is given, the parameter set corresponding to S2 will be used.

This chapter is organized as follows. The genetic algorithm with improved genetic operations will be briefly described in section 2. The specific structure of the proposed dynamic neural network will be presented in section 3. In section 4, a Cantonese-digit speech recognition system will be discussed. The results for recognizing thirteen Cantonese digits and a conclusion will be given in section 5 and 6 respectively.

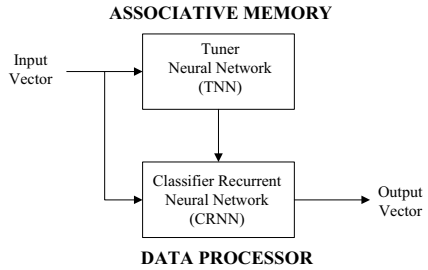
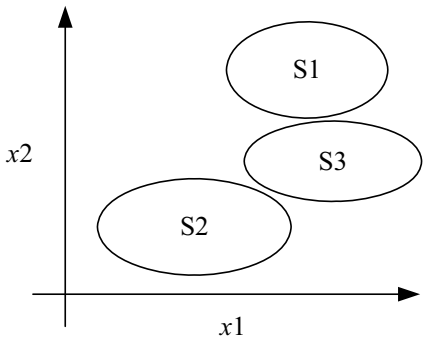
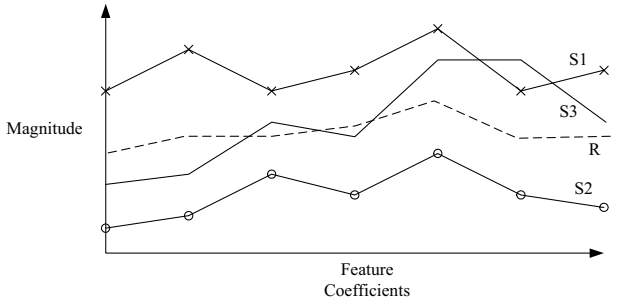


Figure 1. Architecture of the proposed dynamic neural network.



(a)



(b)

Figure 2. (a) Diagram showing 3 data sets in the spatial domain. (b) Diagram showing the feature curves of the 3 sets.

2. Genetic Algorithm with Improved Genetic Operations

Genetic algorithms (GAs) are powerful searching algorithms that can be used to solve optimization problems. The standard GA process (Michalewicz, 1994) is shown in Fig. 3. First, a population of chromosomes is created. Second, the chromosomes are evaluated by a defined fitness function. Third, some of the chromosomes are selected for performing genetic operations. Fourth, the genetic operations of crossover and mutation are performed. The produced offspring replaces their parents in the population. This GA process repeats until a user-defined criterion is reached. In this chapter, the standard GA is modified and new genetic operators are introduced to improve its performance. The improved GA process is shown in Fig. 4. Its details will be given as follows.

```
Procedure of the standard GA
begin
        τ → 0                  // τ: iteration number
initialize P(τ)                //P(τ): population for iteration τ
        evaluate f(P(τ))       // f(P(τ)):fitness function
while (not termination condition) do
      begin
          τ→τ+1
          select 2 parents p1 and p2 from P(τ–1)
          perform genetic operations (crossover and mutation)
          reproduce a new P(τ)
          evaluate f(P(τ))
       end
end
```

Figure 3. Procedure of the standard GA

```
Procedure of the improved GA
begin
        τ → 0                  // τ: iteration number
initialize P(τ)                //P(τ): population for iteration τ
        evaluate f(P(τ))       // f(P(τ)):fitness function
while (not termination condition) do
        begin
        τ→τ+1
        select 2 parents p1 and p2 from P(τ–1)
        perform crossover operation according to equations (7) - (10)
        perform mutation operation according to equation (14) to generate the offspring os
        // reproduce a new P(τ)
                if random number < pa   // pa: probability of acceptance
                        os replaces the chromosome with the smallest fitness value in the
                        population
                 else if f(os) > smallest fitness value in the P(τ–1)
                        os replaces the chromosome with the smallest fitness value
                 end
        evaluate f(P(τ))
        end
end
```

Figure 4. Procedure of the improved GA.

2.1 Initial Population

The initial population is a potential solution set **P**. The first set of population is usually generated randomly.

$$\mathbf{P} = \left\{ \mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_{pop\_size} \right\} \tag{1}$$

$$\mathbf{p}_i = \begin{bmatrix} p_{i_1} & p_{i_2} & \cdots & p_{i_j} & \cdots & p_{i_{no\_vars}} \end{bmatrix},$$

$$i = 1, 2, \ldots, pop\_size; j = 1, 2, \ldots, no\_vars \quad (2)$$

$$para^j_{\min} \le p_{i_j} \le para^j_{\max} \quad (3)$$

where *pop_size* denotes the population size; *no_vars* denotes the number of variables to be tuned; p_{i_j}, $i = 1, 2, \ldots, pop\_size$; $j = 1, 2, \ldots, no\_vars$, are the parameters (genes) to be tuned; $para^j_{\min}$ and $para^j_{\max}$ are the minimum and maximum values of the parameter p_{i_j} respectively for all i. It can be seen from (1) to (3) that the potential solution set **P** contains some candidate solutions $\mathbf{p}_i$ (chromosomes). The chromosome $\mathbf{p}_i$ contains some variables p_{i_j} (genes).

2.2 Evaluation

Each chromosome in the population will be evaluated by a defined fitness function. The better chromosomes will return higher values in this process. The fitness function to evaluate a chromosome in the population can be written as,

$$fitness = f(\mathbf{p}_i) \quad (4)$$

The form of the fitness function depends on the application.

2.3 Selection

Two chromosomes in the population will be selected to undergo genetic operations for reproduction by the method of spinning the roulette wheel (Michalewicz, 1994). It is believed that high potential parents will produce better offspring (survival of the best ones). The chromosome having a higher fitness value should therefore have a higher chance to be selected. The selection can be done by assigning a probability q_i to the chromosome $\mathbf{p}_i$:

$$q_i = \frac{f(\mathbf{p}_i)}{\sum_{k=1}^{pop\_size} f(\mathbf{p}_k)}, \; i = 1, 2, \ldots, pop\_size \quad (5)$$

The cumulative probability $\hat{q}_i$ for the chromosome $\mathbf{p}_i$ is defined as,

$$\hat{q}_i = \sum_{k=1}^{i} q_k, \; i = 1, 2, \ldots, pop\_size \quad (6)$$

The selection process starts by randomly generating a nonzero floating-point number, $d \in [0 \quad 1]$. Then, the chromosome $\mathbf{p}_i$ is chosen if $\hat{q}_{i-1} < d \le \hat{q}_i$ ($\hat{q}_0 = 0$). It can be observed from this selection process that a chromosome having a larger $f(\mathbf{p}_i)$ will have a higher chance to be selected. Consequently, the best chromosomes will get more offspring, the average will stay and the worst will die off. In the selection process, only two chromosomes will be selected to undergo the genetic operations.

2.4 Genetic Operations

The genetic operations are to generate some new chromosomes (offspring) from their parents after the selection process. They include the crossover and the mutation operations.

A. Crossover

The crossover operation is mainly for exchanging information from the two parents, chromosomes $\mathbf{p}_1$ and $\mathbf{p}_2$, obtained in the selection process. The two parents will produce two offspring. First, four chromosomes will be generated according to the following equations,

$$\mathbf{o}_{\mathbf{s}_c}^1 = \left[o_{\mathbf{s}_1}^1 \quad o_{\mathbf{s}_2}^1 \quad \cdots \quad o_{\mathbf{s}_{no\_vars}}^1\right] = \frac{\mathbf{p}_1 + \mathbf{p}_2}{2} \tag{7}$$

$$\mathbf{o}_{\mathbf{s}_c}^2 = \left[o_{\mathbf{s}_1}^2 \quad o_{\mathbf{s}_2}^2 \quad \cdots \quad o_{\mathbf{s}_{no\_vars}}^2\right] = \mathbf{p}_{\max}(1-w) + \max(\mathbf{p}_1, \mathbf{p}_2)w \tag{8}$$

$$\mathbf{o}_{\mathbf{s}_c}^3 = \left[o_{\mathbf{s}_1}^3 \quad o_{\mathbf{s}_2}^3 \quad \cdots \quad o_{\mathbf{s}_{no\_vars}}^3\right] = \mathbf{p}_{\min}(1-w) + \min(\mathbf{p}_1, \mathbf{p}_2)w \tag{9}$$

$$\mathbf{o}_{\mathbf{s}_c}^4 = \left[o_{\mathbf{s}_1}^4 \quad o_{\mathbf{s}_2}^4 \quad \cdots \quad o_{\mathbf{s}_{no\_vars}}^4\right] = \frac{(\mathbf{p}_{\max} + \mathbf{p}_{\min})(1-w) + (\mathbf{p}_1 + \mathbf{p}_2)w}{2} \tag{10}$$

$$\mathbf{p}_{\max} = \left[para_{\max}^1 \quad para_{\max}^2 \quad \cdots \quad para_{\max}^{no\_vars}\right] \tag{11}$$

$$\mathbf{p}_{\min} = \left[para_{\min}^1 \quad para_{\min}^2 \quad \cdots \quad para_{\min}^{no\_vars}\right] \tag{12}$$

where $w \in [0 \quad 1]$ denotes a weight to be determined by users, $\max(\mathbf{p}_1, \mathbf{p}_2)$ denotes the vector with each element obtained by taking the maximum among the corresponding element of p1 and p2. For instance, $\max([1 \quad -2 \quad 3], [2 \quad 3 \quad 1]) = [2 \quad 3 \quad 3]$. Similarly, $\min(\mathbf{p}_1, \mathbf{p}_2)$ gives a vector by taking the minimum value. For instance, $\min([1 \quad -2 \quad 3], [2 \quad 3 \quad 1]) = [1 \quad -2 \quad 1]$. Among $\mathbf{o}_{\mathbf{s}_c}^1$ to $\mathbf{o}_{\mathbf{s}_c}^4$, the two with the largest fitness value are used as the offspring of the crossover operation. These two offspring are put back into the population to replace their parents. One of the offspring is defined as,

$$\mathbf{o}_{\mathbf{s}} \equiv \left[o_{s_1} \quad o_{s_2} \quad \cdots \quad o_{s_{no\_vars}}\right] = \mathbf{o}_{\mathbf{s}_c}^{i_{os}} \tag{13}$$

i_{os} denotes the index i which gives a maximum value of $f\left(\mathbf{o}_{\mathbf{s}_c}^i\right)$, i = 1, 2, 3 ,4.

If the crossover operation can provide a good offspring, a higher fitness value can be reached in a smaller number of iteration. In general, two-point crossover, multipoint crossover, arithmetic crossover or heuristic crossover can be employed to realize the crossover operation (Michalewicz, 1994). The offspring generated by these methods, however, may not be better than that from our approach. As seen from (7) to (10), the potential offspring after the crossover operation spreads over the domain. While (7) and (10) result in searching around the centre region of the domain (a value of w near to 1 in (10) can move $\mathbf{o}_{\mathbf{s}_c}^4$ to be near $\frac{\mathbf{p}_1 + \mathbf{p}_2}{2}$), (8) and (9) move the potential offspring to be near the

domain boundary (a small value of w in (8) and (9) can move $\mathbf{o}_{\mathbf{s}_c}^2$ and $\mathbf{o}_{\mathbf{s}_c}^3$ to be near $\mathbf{p}_{\max}$ and $\mathbf{p}_{\min}$ respectively).

B. Mutation

The offspring (13) will then undergo the mutation operation, which changes the genes of the chromosome. Consequently, the features of the chromosomes inherited from their parents can be changed. In general, various methods like boundary mutation, uniform mutation or non-uniform mutation (Michalewicz, 1994) can be employed to realize the mutation operation. Boundary mutation is to change the value of a randomly selected gene to its upper or lower bound. Uniform mutation is to change the value of a randomly selected gene to a value between its upper and lower bounds. Non-uniform mutation is capable of fine-tuning the parameters by increasing or decreasing the value of a randomly selected gene by a weighted random number. The weight is usually a monotonic decreasing function of the number of iteration. We propose a different process of mutation with details as follows. Every gene of the offspring os of (13) will have a chance to mutate governed by a probability of mutation, $p_m \in [0 \quad 1]$, which is defined by the user. This probability gives an expected number ($p_m \times$ no_vars $\times$ pop_size) of genes that undergo the mutation. For each gene, a random number between 0 and 1 will be generated such that if it is less than or equal to p_m, the operation of mutation will take place on that gene and updated instantly. The gene of the offspring of (13) is then mutated by:

$$\hat{o}_{s_k} = \begin{cases} o_{s_k} + \Delta o_{s_k}^U \text{ if } f(\mathbf{o}_\mathbf{s} + \Delta\mathbf{o}_{\mathbf{s}_k}^U) \geq f(\mathbf{o}_\mathbf{s} - \Delta\mathbf{o}_{\mathbf{s}_k}^L) \\ o_{s_k} - \Delta o_{s_k}^L \text{ if } f(\mathbf{o}_\mathbf{s} + \Delta\mathbf{o}_{\mathbf{s}_k}^U) < f(\mathbf{o}_\mathbf{s} - \Delta\mathbf{o}_{\mathbf{s}_k}^L) \end{cases}, k = 1, 2, \ldots, no\_vars \tag{14}$$

where

$$\Delta o_{s_k}^U = w_{m_k} r\left(para_{\max}^k - o_{s_k}\right) \tag{15}$$

$$\Delta o_{s_k}^L = w_{m_k} r\left(o_{s_k} - para_{\min}^k\right) \tag{16}$$

$$\Delta\mathbf{o}_{\mathbf{s}_k}^U = \begin{bmatrix} 0 & 0 & \cdots & \Delta o_{s_k}^U & \cdots & 0 \end{bmatrix} \tag{17}$$

$$\Delta\mathbf{o}_{\mathbf{s}_k}^L = \begin{bmatrix} 0 & 0 & \cdots & \Delta o_{s_k}^L & \cdots & 0 \end{bmatrix} \tag{18}$$

$r \in [0 \quad 1]$ is a randomly generated number; $w_{m_k} \in (0 \quad 1]$ is a weight governing the magnitudes of $\Delta o_{s_k}^U$ and $\Delta o_{s_k}^L$. The value of weight w_{m_k} is varied by the value of $\frac{\tau}{T}$ to serve a fine-tuning purpose. T is the total number of iteration. In order to perform a local search, the value of weight w_{m_k} should be very small as $\frac{\tau}{T}$ increases in order to reduce the significance of the mutation. Under this assumption, a monotonic decreasing function governing w_{m_k} is proposed to be,

$$w_{m_k} = w_f \left(1 - \frac{\tau}{T}\right)^{\frac{1}{w_\tau}} \geq 0 \tag{19}$$

where $w_f \in [0 \quad 1]$ and $w_\tau > 0$ are variables to be chosen to determine the initial value and the decay rate respectively. For a large value of w_f, it can be seen from (15) and (16) that $\Delta o_{s_k}^U \approx r\left(para_{\max}^k - o_{s_k}\right)$ and $\Delta o_{s_k}^L \approx r\left(o_{s_k} - para_{\min}^k\right)$ initially as $\left(1 - \frac{\tau}{T}\right)^{\frac{1}{w_\tau}} \approx 1$, which ensure a large search space. When the value of $\left(1 - \frac{\tau}{T}\right)^{\frac{1}{w_\tau}} \approx 0$, it can be seen that the values of $\Delta o_{s_k}^U$ and $\Delta o_{s_k}^L$ are small to ensure a small search space for fine-tuning.

2.5 Reproduction

The reproduction process takes place after the genetic operations. The new offspring will be evaluated using the fitness function of (4). This new offspring will replace the chromosome with the smallest fitness value among the population if a randomly generated number within 0 to 1 is smaller than $p_a \in [0 \quad 1]$, which is the probability of acceptance defined by users. Otherwise, the new offspring will replace the chromosome with the smallest fitness value only if the fitness value of the offspring is greater than the fitness value of that chromosome in the population. p_a is effectively the probability of accepting a bad offspring in order to reduce the chance of converging to a local optimum. Hence, the possibility of reaching the global optimum is kept.

After the operation of selection, crossover, mutation and reproduction, a new population is generated. This new population will repeat the same process. Such an iterative process can be terminated when the result reaches a defined condition, e.g. the change of the fitness values between the current and the previous iteration is less than 0.001, or a defined number of iteration has been reached.

3. Modified Dynamic Neural Network

Recurrent Neural Network (RNN) is a dynamic network which is commonly used to tackle complex dynamic sequential problems, such as time series prediction, dynamic system identification and grammatical inference. With the specific network structure, the temporal information can be brought to the next time frame. As a result, the RNN can process static as well as time-varying information. Elman network, Jordan network (Rabiner & Juang, 1993), etc. are commonly used RNNs. They are constructed as closed-loop systems while the hidden node outputs or the network outputs are fed back to the network inputs as the dynamic information respectively. However, training a recurrent network to the desired behavior is not easy. It is because there is only a fix set of parameters representing both the temporal and static features of the input data sets.

In order to improve the classifying ability of a traditional RNN, a traditional 3-layer feed-forward neural network and a recurrent neural network can be combined together as a modified network. A traditional 3-layer feed-forward neural network provides a distinct solution to static pattern classification problems (Jang, 1997) and a recurrent network

provides a solution to time-varying problems. The proposed neural network architecture employs a feed-forward neural network (rule-base neural network) to offer some rule information to the recurrent neural network (classifier recurrent neural network) with respect to the input patterns. Thus, the additional information provided by the rule-base neural network can compensate the limitation of the fixed parameter sets.

3.1 Model of the Modified Dynamic Neural Network

The proposed modified dynamic neural network consists of two parts, namely the Tuner Neural Network (TNN) and the Classifier Recurrent Neural Network (CRNN). The network architecture is shown in Fig.5.

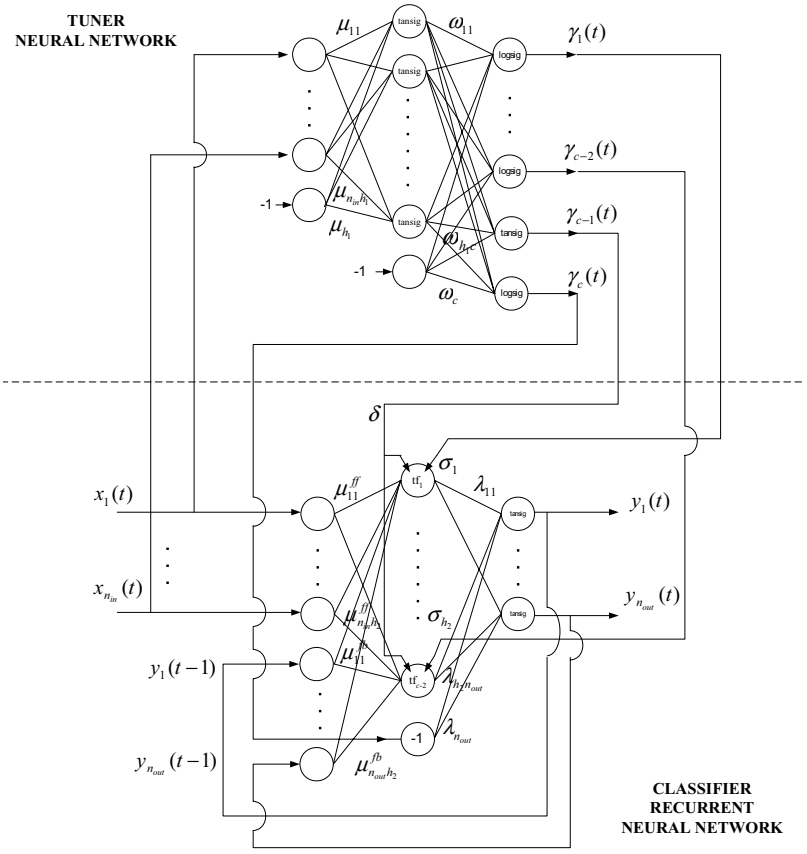


Figure 5. Modified dynamic neural network architecture.

A. Tuner Neural Network

The tuner neural network (TNN) is a traditional 3-layer feed-forward neural network that provides suitable parameters to the classifier recurrent neural network according to the input patterns. The input-output relationship of the TNN is defined as follows.

$$\gamma_g(t) = logsig\left[\sum_{j=1}^{h_1}\omega_{jg}\, tansig\left(\sum_{i=1}^{n_{in}}\mu_{ij}x_i(t)-\mu_j\right)-\omega_g\right] \tag{20}$$

$$\gamma_{c-1}(t) = tansig\left[\sum_{j=1}^{h_1}\omega_{jc-1}\, tansig\left(\sum_{i=1}^{n_{in}}\mu_{ij}x_i(t)-\mu_j\right)-\omega_{c-1}\right] \tag{21}$$

$$\gamma_c(t) = logsig\left[\sum_{j=1}^{h_1}\omega_{jc}\, tansig\left(\sum_{i=1}^{n_{in}}\mu_{ij}x_i(t)-\mu_j\right)-\omega_c\right] \tag{22}$$

where $\gamma_g(t)$, $\gamma_{c-1}(t)$, and $\gamma_c(t)$ are the outputs of the TNN for the t-th data frame; $tansig(\alpha)=\frac{2}{1+e^{-2\alpha}}-1$, $\alpha\in\Re$, is the tangent sigmoid function; $\log sig(\alpha)=\frac{1}{1+e^{-\alpha}}$, $\alpha\in\Re$, denotes the logarithmic sigmoid function; ω_{jg}, j = 1, 2, … ,h_1, denotes the weight of the link between the j-th hidden node and the g-th output; h_1 is a non-zero positive integer denoting the number of hidden nodes; μ_{ij}, i = 1, 2, …, n_{in}, denotes the weight of the link between the i-th input node and the j-th hidden node; $x_i(t)$ denotes the i-th input of the TNN; μ_j, denotes the weight of the j-th bias term for the hidden layer; ω_g, ω_{t-1}, and ω_t are the link weights of the bias terms for the output layer; n_{in} and c denote the numbers of input and output respectively.

B. Classified Recurrent Neural Network

As shown in Fig. 5, the classifier recurrent neural network (CRNN) is a 3-layer recurrent network. It analyses the input patterns, recurrent information and the information provided by the TNN to produce the network outputs. The input patterns are tokens of short-time feature frames. The feature frames will be fed to the input of the network one-by-one. The outputs of the network will be fed back to the network inputs with one sampling-period delay. The input-output relationship of the CRNN is defined as follows.

$$y_k(t) = tansig\left[\sum_{j=1}^{h_2}\lambda_{jk}\, tf\left(\sum_{i=1}^{n_{in}}\mu_{ij}^{ff}x_i(t)+\sum_{q=1}^{n_{out}}\mu_{qj}^{fb}y_q(t-1)\right)-\lambda_k\gamma_c(t)\right],\ k = 1, 2, \ldots, n_{out} \tag{23}$$

where $y_k(t)$ denotes the k-th output of the CRNN for the t-th data frame; h_2 is a non-zero positive integer denoting the number of hidden nodes (excluding the bias node); λ_{jk}, j = 1,2, … ,h_2 denotes the weight of the link between the j-th hidden node and the k-th output; μ_{ij}^{ff}, i = 1,2, … ,n_{in} denotes the weight of the link between the i-th input and the j-th hidden node; $\gamma_c(t)$ denotes the last output from the TNN; μ_{qj}^{fb}, q = 1, 2, …, n_{out} denotes the weight of the link between the q-th recurrent input and the j-th hidden node; λ_k denotes the link weight of the k-th bias term for the output layer; $tf(\cdot)$ denotes the hidden node activation function and is defined as follows.

$$tf(\chi_g;\delta,\sigma_g) = \frac{2}{1+e^{\frac{-(\chi_g-\delta)}{2\sigma_g^2}}}-1,\ g = 1,2, \ldots, h_2 \tag{24}$$

χ_g, δ and σ_g denote the input and the two parameters of the activation function respectively. It should be noted that δ and σ_g are the outputs of the TNN. Thus, $\sigma_g = \gamma_g$, g = 1, 2, ..., h_2;

$h_2 = c - 2$, and $\delta = \gamma_c - 1$. These parameters are given by the TNN to guide the CRNN how to handle the input data. Owing to the dynamic structure of the proposed NN, both the static and dynamic properties of the data can be modeled. Different parameter values change the shape of the non-linear activation function (24). The CRNN performs dynamic adaptation to the input frame variations through the recurrent links. Each frame pattern will have its own parameter set. Owing to the structure of the modified RNN, both static (TNN) and dynamic (CRNN) changes of the data are classified at the same time.

C. Training of the Modified Dynamic Neural Network

An individual class network training process is employed. Every data class has its own network, where every network has the same structure. The training process is to train the network input frame(s) to match the network temporal output frame(s); where the output frame(s) of the network is equal to the network input frame(s). Thus, the highest fitness value can be obtained from the network by using the same class of input data set. All parameters in the modified RNN are trained by the improved GA (Lam et al., 2004).

4. Speed Recognition System

Electronic Book (eBook) provides a new trend of reading. By using eBook, traditional reading materials can be enriched. However, the input method for an eBook reader is typically realized by handwritten graffiti recognition through the touch-screen. In order to develop a more natural way for eBook inputs, speech recognition is proposed.

Speech recognition is a tool to make machines understand the sounds made by the human vocal tract, which is called speech. Speech is an acoustic signal that varies in time. Every speech has its unique characteristics in the frequency domain, called speech features. Although a speech frame is characterized by unique features, other factors such as background noise, words with similar spectral characteristics (especially common for Cantonese words) may affect the recognition accuracy. A good feature extraction method and a high-accuracy classification algorithm are needed to produce a high-performance speech recognizer.

Cantonese speech composes of a chain of mono-syllabic sound. Each Cantonese-character speech is a combined unit of a tone and a syllable. There are nine possible tones for a Cantonese syllable. Some Cantonese characters share the same vowel, e.g. the Cantonese character of "1" (/jat1/) and the Cantonese character of "7" (/cat1/) share the same vowel of /a/ (Markowitz, 1996). It is a difficult task to recognize Cantonese characters that requires the discrimination of not only characters with different syllable but also tones of the same syllable with high accuracy.

In general, speech recognition (Rabiner & Juang, 1993) is realized in two stages: speech preprocessing and classification. The preprocessing stage involves segmentation and feature extraction. Segmentation is used to define the boundaries of the temporal speech segments that represent the basic phonetic components. Then, the stationary properties of the individual segment can be modeled by some static classification approach. The dynamic properties of the temporal segments can be modeled by some dynamic classification approach. Feature extraction is a technique to find a good representation to a speech signal. Normally, the time-domain speech signals will be windowed into speech frames. From each speech frame, the fast Fourier transform is applied to obtain the frequency spectrum. Based

on the frequency spectrum, digital signal analysis techniques will be applied to obtain the cepstral coefficients, which describe the features of the speech frame. Filter-bank analysis (Rabiner & Juang, 1993) is often used to analyze the speech frames for extracting features. By distributing different band-pass filters in the mel-scale of frequency, which models the characteristics of the human ears, the frames of speech feature coefficients can be obtained. Using the feature coefficients, we can perform the second step of classification..

Speech classification methods can be categorized into 3 types: template matching, acoustic-phonetic recognition and stochastic processing. For the template matching technique, reference speech units called templates are used to perform the matching process between the testing speech units and the templates. Thus, the testing speech units that produced close matching with the reference templates can be identified. The acoustic-phonetic recognition approach is a phoneme level speech recognition approach. By using this approach, the acoustic similarity among the phoneme combinations in a speech will be used to identify the input speech. The stochastic process for speech recognition is similar to the template matching process that requires the reference speech units for identifying the input speech. The main difference is that the stochastic process performs a statistical and probabilistic analysis matching process, which is not a direct mapping to the reference templates.

Two popular Cantonese speech recognition techniques are that using the Hidden Markov Model (HMM) and that using the Neural Network (NN) (Rabiner & Juang, 1993; Wu & Chan, 1993; Lee et al., 1998). HMM is one well-known statistical state-sequence recognizing approach for speech recognition (Markowitz, 1996). The states in a HMM structure represent the stochastic process, and the directional links between states indicate the transitions of flowing from one state to another. With a different states-and-transitions structure for each speech pattern, recognition can be done by matching the testing speech unit to the reference models. The Baum-Welch maximum-likelihood algorithm can be employed to compare the probability score between the testing and reference models as the likelihood index. Another verification process for HMM speech recognition is done by the Viterbi algorithm, which is a method to determine the nodes and sequence of the testing speech unit that closely match the reference models.

Recently, neural networks (NNs), especially the recurrent neural networks, are commonly used for speech recognition. Based on its connectionist modeling technique, non-linear functions can be modeled. Thanks to the capability of being a universal approximator through training, an NN can be trained to become a classifier for a certain input pattern.

On applying NNs to realize the classification of speech patterns, we can adopt a static or a dynamic pattern classification method. The static pattern classification method uses a conventional 3-layer feed-forward neural network to model each single-character Cantonese speech. The static properties of the speech frames can be analyzed. However, the input vector of the network contains no acoustic feature, and the network is only suitable for recognizing speech patterns with a single syllable.

Since the dynamic properties between speech frames are important for recognizing speech, we should consider using recurrent neural networks (RNNs) as the classifiers, and learn the relationships between speech frames through the training process. In practice, an RNN is suitable to classify speech with multiple syllables. However, an RNN consumes more computing power than a feed-forward NN. Owing to the recurrent information fed back, an

RNN needs a large number of network parameters so as to cope with the speech frame updates and the recurrent information updates for each speech pattern. As the number of parameters is large, a longer training time is needed for an RNN to converge.
It should be noted that one neural network, feed-forward or recurrent, is needed effectively to model one speech pattern. If we have to recognize a large number of recognition vocabularies, a standard structured NN is difficult to achieve a good performance. It is because the trained network parameters are used to model the features of the input patterns. When the number of input patterns is large and they are not sufficiently clustered into the same class, more network parameters are needed to model them. A complicated network structure will result. This will degrade the performance of the network in terms of computational power, convergence rate and recognition accuracy.

A. Speech feature extraction

The speech patterns are formatted in 8-bit PCM format sampled at 11kHz. The obtained time-domain speech vector $\mathbf{s} = [s(1) \;\; s(2) \;\; \cdots \;\; s(no\_sample)]$ is then windowed by the 128-sample sliding Hamming windows $wh(\tau)$ with 50% overlap to form speech frames. The zero-crossing rate and speech energy have to be found in order to determine whether the frames are voiced (sonorant) or unvoiced (non-sonorant). A speech frame with a high zero-crossing rate but low speech energy level is defined as the non-sonorant speech frame. Conversely, a sonorant speech frame will have a high speech energy level but low zero-crossing rate. The process is defined as follows.

$$E_n = \sum_{\tau=1}^{128} s(m_n + \tau)^2 \quad ,n = 1, 2, \ldots, no\_frame \tag{25}$$

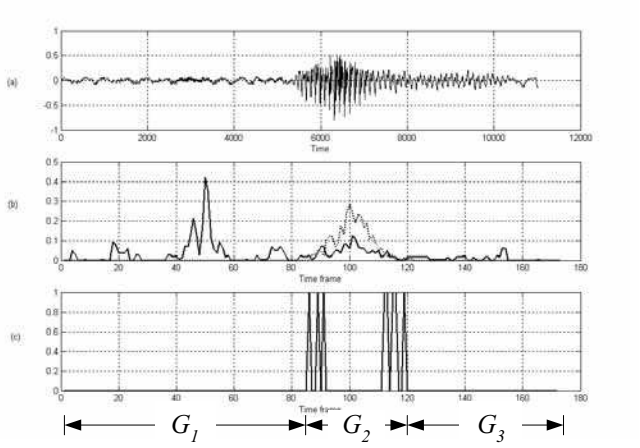
$$Z_n = \frac{1}{2\times 128} \sum_{\tau=1}^{127} \left[s_s(m_n + \tau) - s_s(m_n + \tau + 1) \right], \quad n = 1, 2, \ldots, no\_frame \tag{26}$$

$$m_n = [64(n-1)] \quad ,n = 1, 2, \ldots, no\_frame \tag{27}$$

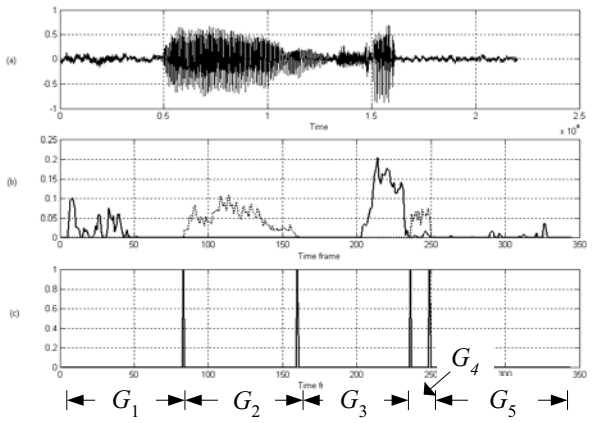
$$s_s(\tau) = \begin{cases} 1, \text{when } s(\tau) >= 0 \\ -1, \text{when } s(\tau) < 0 \end{cases}, \quad \tau=1, 2, \ldots, no\_sample \tag{28}$$

where E_n denotes the speech energy of the n-th frame; m_n denotes the starting index of the n-th frame; *no_frame* denotes the number of frames for a speech signal; Z_n denotes the zero-crossing rate of the n-th frame; *no_sample* (= 128) denotes the number of time samples of a speech frame (i.e. the number elements of a speech frame vector). As shown in Fig. 6(a), G_1, G_2 and G_3 indicate the non-sonorant, sonorant and non-sonorant region of a single-syllable Cantonese-digit speech. The starting speech frame of a new group begins when the speech energy curve and the zero-crossing curve intersect, provided that the group has more than six speech frames (i.e. a duration longer than 40ms); otherwise, no group change will take place. Therefore, the speech signal as shown in Fig. 6(a) gets three groups. Five groups are obtained from a double-syllable Cantonese-number speech as shown in Fig. 6(b). G_1, G_3 and G_5 are the non-sonorant regions due to high zero-crossing rate and low speech energy level.

While G_2 and G_4 are the sonorant regions due to high speech energy level and low zero-crossing rate. Since a pause has been added after the first syllable, G_3 is obtained.



(a)



(b)

Figure 6. Acoustic feature groups of (a) single Cantonese digit, (b) double Cantonese digits.

Fast Fourier Transform (FFT) and uniform filter-bank filtering are then performed to each speech frame. The feature coefficients of each speech frame are defined as follows.

$$\mathbf{Sf}_n = \left[Sf_n(1) \quad Sf_n(2) \quad \cdots \quad Sf_n(128)\right] = Re\left\{FFT\left(\left[wh(\tau)s_n(\tau)\right]\right)\right\},\ 1 \le \tau \le 128 \tag{29}$$

$$c_n^{\beta} = \frac{20\log_{10}\sum_{l=1}^{\alpha} Sf_n(\rho_\beta + l)wt(\rho_\beta + l)}{\alpha} \quad \beta = 1,2, \ldots, no\_filter \tag{30}$$

$$wt(l) = \begin{cases} \dfrac{2l-1}{\alpha}, & 1 \le l \le \dfrac{\alpha}{2} \\ \dfrac{2(\alpha - l + 1)}{\alpha}, & \dfrac{\alpha}{2} + 1 \le l \le \alpha \end{cases} \tag{31}$$

$$\alpha = floor\left(\frac{128}{no\_filter}\right) \times 2 \tag{32}$$

$$\rho_\beta = 0.5\alpha(\beta - 1)\,,\ \beta = 1, 2, \ldots, no\_filter \tag{33}$$

$$\mathbf{D}_n = \left[c_n^2 - c_n^1 \quad c_n^3 - c_n^2 \quad \cdots \quad c_n^{no\_filter} - c_n^{no\_filter-1}\right] \tag{34}$$

where $\mathbf{Sf}_n$ denotes the frequency spectrum of the n-th speech frame; $Re\{\cdot\}$ denotes the real part of the argument vector; $FFT(\cdot)$ denotes the fast Fourier transform function; $s_n = \left[s_n(1) \quad s_n(2) \quad \cdots \quad s_n(128)\right]$ denotes the n-th speech frame in time-domain; $wt(\cdot)$ denotes the triangular window; α denotes the number of frequency components tackled by each band-pass filter; $floor(\cdot)$ denotes the floor function which is used to round up a floating point number; *no_filter* denotes the number of band-pass filters; ρ_β denotes the starting index of the β-th band-pass filter; c_n^{β} denotes the mean power output from the β-th band-pass filter for the n-th frame; $\mathbf{D}_n$ is a vector formed by the magnitude differences between two consecutive band-pass filter outputs.

The neighboring speech frames of the same nature, i.e. voiced/sonorant or unvoiced/non-sonorant frames, will be grouped together as shown in Fig. 6. The mean feature coefficient of all the speech frames in the same group will be calculated as follows.

$$\mathbf{scoeff}_\eta = \frac{\sum_{n=1}^{G_\eta} \mathbf{D}_n}{G_\eta}\,,\ \eta = 1,2, \ldots, no\_group \tag{35}$$

where $\mathbf{scoeff}_\eta$ denotes η-th speech feature group; G_η denotes the number of speech frames in the η-th group; *no_group* denotes the number of groups that can be segmented from the speech. Thus, the acoustic feature sequence of the speech can be written as $\mathbf{Sp} = \left[\mathbf{scoeff}_1 \quad \mathbf{scoeff}_2 \quad \cdots \quad \mathbf{scoeff}_{no\_group}\right]$, and each element vector is then normalized as input to the dynamic variable-parameter neural network in sequence to do the speech recognition.

B. Speech classification

The inputs of the modified RNN is defined as $\mathbf{x} = \dfrac{\mathbf{Sp}}{\|\mathbf{Sp}\|}$ where $\|\cdot\|$ denotes the l_2 vector norm.

The objective of the training process for each network is to adjust the parameters so as to minimize the error between the network outputs and the desired values, where the desired values are the inputs of the network. The performance of the network is governed by the value of fitness, which is a function the error value *err* :

$$fitness = \frac{1}{1+err} \tag{36}$$

$$err = \mathbf{e}_w sort\left(\left[\sum_{\eta=1}^{ng}\left|d_\eta^1 - y_\eta^1\right| \quad \sum_{\eta=1}^{ng}\left|d_\eta^2 - y_\eta^2\right| \quad \cdots \quad \sum_{\eta=1}^{ng}\left|d_\eta^{n_{out}} - y_\eta^{n_{out}}\right|\right]^{\mathrm{T}}\right) \tag{37}$$

$$\mathbf{e}_w = \left[e_w^1 \quad e_w^2 \quad \cdots \quad e_w^k\right] = \left[\frac{1}{n_{out}} \quad \frac{2}{n_{out}} \quad \cdots \quad 1\right] \tag{38}$$

where *err* is governed by the sum absolute error between the desired output $\mathbf{d}_\eta = \left[d_\eta^1 \quad d_\eta^2 \quad \cdots \quad d_\eta^{n_{out}}\right]$ and the network output $\mathbf{y}_\eta = \left[y_\eta^1 \quad y_\eta^2 \quad \cdots \quad y_\eta^{n_{out}}\right]$; $\mathbf{e}_w$ denotes the error-weight vector, *sort*(·) returns a vector that has the argument vector's elements sorted in descending order; *ng* denotes the number of groups that can be segmented from the speech signal. It should be noted that the desired value will be equal to the input in the classification process. The fitness value will be optimized by the improved GA.

After the training process of each network has been done, *m* sets of parameters are obtained. In order to classify the target class to which an input pattern belongs, the input pattern will be tested simultaneously by all the networks. Therefore, *m* fitness values will be produced by the networks according to (36). The class of the network with the highest fitness value is the most likely class to which the input pattern belongs. In other words, the most likely word class is given by

$$dig_{\max} = \underset{1<dig<no\_pat}{\operatorname{argmax}} \; fitness(\mathbf{w}_{dig}, \mathbf{x}_\eta) \tag{39}$$

where *dig* denotes the word class; *no_pat* denotes the number of word classes; *fitness*(·) denotes the fitness value offered by the proposed NN with input $\mathbf{x}_\eta$ and the *dig*-th network weight set $\mathbf{w}_{dig}$.

5. Simulation Results

The Cantonese-digit speech recognition is processed as shown in Fig. 7. Speech signals are recorded from a male speaker with a low cost microphone. Eleven single Cantonese digits (0 to 10) and two double Cantonese digits (12 and 20) are used to test the performance of the proposed network, where the double Cantonese digits (12 and 20) are mainly used to test the sequential classification ability of the proposed recurrent network. The Cantonese syllables for the single digits can be found in "http://humanum.arts.cuhk.hk/Lexis/Canton/". The syllables of the two double

Cantonese digits (12 and 20) are the syllable combinations of (10, 2) and (2, 10) respectively. Each Cantonese digit is recorded with 20 repetitions as training data sets and 50 repetitions as testing data sets. The speech signals are then passed to the feature extractor.

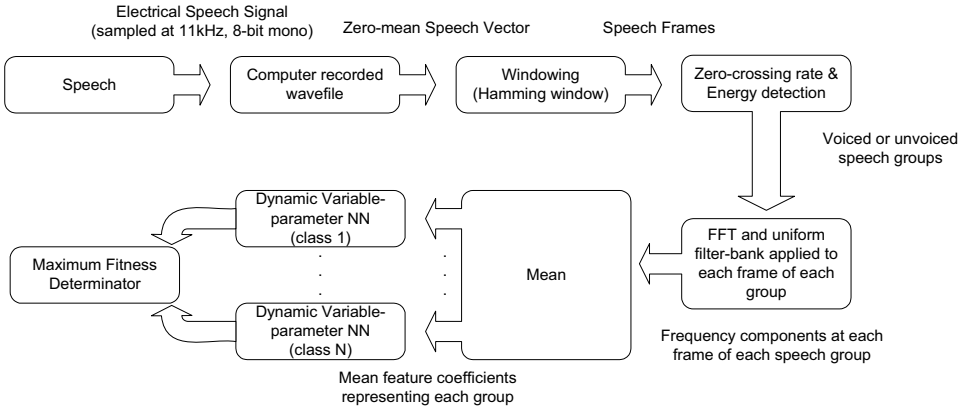


Figure 7. Block diagram of the Cantonese speech recognition system.

Firstly, the time domain speech signal will be windowed by the 128 hamming sliding window with 50% overlap. Zero-crossing rate and average energy of the speeches are calculated to obtain the acoustic speech frames using (25) to (28). If a speech frame has over 30% of its maximum zero-crossing rate and the average energy is less than 10% of its maximum energy level, the frame will be regarded as non-sonorant; otherwise, it is regarded as sonorant. FFT is then employed to obtain the frequency spectrum of each speech frame and the feature coefficients are extracted by using a uniform filter-bank. 12 feature coefficients will be obtained for each group using (29) to (34). Finally, the element vectors of the normalized acoustic feature sequence (**Sp**) will become the inputs of the modified RNN.

One network will be used to model one input class on doing the Cantonese digits recognition. The proposed network is a 24-inputs-12-outputs modified recurrent neural network. The element vectors of acoustic feature sequence are fed to the modified RNN one-by-one. Each group will have 12 parameters to represent the temporal speech feature. Hence, the network inputs are 12 feature coefficients given from the current speech frame and 12 parameters fed back from the outputs of the network at the previous state. The number of recurrent loop(s) is determined by the number of acoustic feature group(s) of the speech signal. The initial feedback values of the recurrent network are set to zeros. The improve GA is used to train all the linked weights, $\begin{bmatrix} \mu_{ij} & b_{h_i} & \omega_{jg} & \mu_{ij}^{ff} & \mu_{qj}^{fb} & \lambda_{jk} \end{bmatrix}$ for all i, j, g, q, k, of the modified RNN. The vector of the linked weights is the chromosome of the improved GA. The control parameters w, p_m, w_f and w_σ for the improved GA are chosen to be 0.5, 0.02, 0.5 and 1 respectively. The upper and lower bound of the chromosome are 2.0 and –2.0 respectively. The number of iteration used for training is 2000. The initial values of

the chromosomes are generated randomly. The number of hidden nodes used for the simulation were (3, 5), (5, 5), (5, 10), (10, 10), (10, 12) and (12, 12) where the first number inside the bracket is the number of hidden nodes used in the TNN and the second number inside the bracket is the number of hidden nodes used in the CRNN. The learning process is done by a personal computer with Pentium 4, 1.4GHz CPU and 256MB RAM. The simulation results of the 11 single-syllable Cantonese digits are tabulated in Table 1. The performance of sequence recognition is tabulated in Table 2. The overall performance of the proposed method is summarized in Table 3.

h_1,h_2		3,5	5,5	5,10	10,10	10,12	12,12
No. of parameters		259	299	509	634	728	782
Digit	Syllable	Performance (%)					
0 (零)	ling4	84	98	92	88	96	84
1 (一)	jat1	94	96	100	100	98	98
2 (二)	ji6	98	100	94	100	100	100
3 (三)	saam1	92	88	88	90	86	86
4 (四)	sei3	100	100	100	100	100	100
5 (五)	ng5	98	100	98	100	100	100
6 (六)	luk6	92	92	94	100	98	88
7 (七)	cat1	90	86	90	90	86	86
8 (八)	baat3	86	80	72	94	90	68
9 (九)	gau2	100	92	96	92	100	92
10 (十)	sap6	64	66	86	96	78	72

Table 1. Recognition performance of 11 Cantonese digits "0" to "10" (test data) by the proposed approach.

h_1,h_2		3,5	5,5	5,10	10,10	10,12	12,12
Digit	Syllable	Performance (%)					
2 (二)	ji6	98	100	94	100	100	100
10 (十)	sap6	64	66	86	96	78	72
12 (十二)	sap6, ji6	98	90	92	98	86	78
20 (二十)	ji6, sap6	88	90	98	94	98	96

Table 2. Recognition performance of the sequential Cantonese digits "2", "10", "12" and "20" (test data) by the proposed approach.

h_1,h_2	3,5	5,5	5,10	10,10	10,12	12,12
Testing Performance (%)	91.1	90.6	92.3	95.5	93.5	88.3
Training Performance (%)	98.5	99.6	98.8	99.2	99.2	98.8

Table 3. Overall recognition performance of 13 testing and training Cantonese digits by the proposed approach.

The results show that the proposed method provides a fast convergence rate. Only 2000 times of iteration for each Cantonese digit are sufficient for the training process to obtain about 99% accuracy models as shown in Table 3. This indicates that the structure of the TNN can provide useful information for the CRNN in order to reduce the number of iteration for training the network. In order to examine the effects of the network size to the network performance, we have tried different combinations of h1 and h2 values. Referring to the results shown in Tables 1 and 3, the best result is obtained when the numbers of hidden nodes (h_1, h_2) are set to (10,10). The recognition accuracy is 95.5%. The number of parameters used in each network at that setting is only 634. Apart from that, if the number of parameters used in each network is reduced to 259, i.e. 41% of the best setting, the recognition accuracy can still reach over 91%. The results demonstrate that the proposed network can use much fewer parameters for a drop of only 4% recognition accuracy. Considering the performance of the network to some Cantonese digits that have same vowel /a/ or /u/ (Lee et al., 1995), the Cantonese digits "1", "7", "8", "10" have the common vowel /a/ and "6", "9" have the common vowel /u/. By using the proposed method, they can be discriminated among each other with accuracy over 90% and 92% respectively. However, if the numbers of hidden nodes are set to (10, 12) and (12, 12), the accuracies of the network are then dropped to 93.5% and 88.3% respectively. This is because the numbers of parameters used for these two settings are increased to 728 and 782 respectively. Hence, there may be some redundant parameters affecting the network accuracy. Referring to Table 2, the sequence of the speech signals can be well determined by the propose method. By using the configuration that produces the best recognition accuracy from Table 1, the recognition accuracies of the four Cantonese digits "2", "10", "12", "20" are 100%, 96%, 98% and 94% respectively. Thus, the proposed network can produce high recognition accuracy for multi-syllable speech patterns although their syllables are similar. As a result, the static and dynamic feature of the Cantonese digits can be obtained effectively and classified clearly by the proposed method. The proposed recognition system is compared with one using a 3-layer fully connected neural network with recurrent paths as shown in Fig 8. The recognition system is trained by the improved GA with the same number of iteration for the proposed system. Besides that, the system is trained and tested by the same data patterns. The results produced by this network are tabulated from Table 4 to Table 6. As shown from the results, the recognition system produces the highest accuracy when the number of hidden node is equal to 22 (804 parameters). The overall recognition accuracy of this setting is 95.4%. By comparing the best performance of this recognition system to the proposed system in terms of the number of parameters used and the recognition accuracy, it can be seen that the proposed system requires a smaller number of parameters but offers similar recognition accuracy. It illustrates that the proposed network architecture can improve the

recognition ability of the Cantonese-digit speech recognition application. In addition, the associative memory technique can improve the learning ability of the recurrent neural network from 98.8% (Table 6) to 99.2% (Table 3).

h_1		8	12	14	16	18	22
No. of parameters		300	444	516	588	660	804
Digit	Syllable	Performance (%)					
0 (零)	ling4	92	92	96	94	96	96
1 (一)	jat1	94	88	98	96	88	94
2 (二)	ji6	98	100	100	100	100	100
3 (三)	saam1	78	84	76	86	90	94
4 (四)	sei3	96	100	98	98	100	100
5 (五)	ng5	100	98	100	92	92	100
6 (六)	luk6	96	96	94	92	96	98
7 (七)	cat1	90	86	94	88	88	96
8 (八)	baat3	88	76	86	96	82	92
9 (九)	gau2	96	84	100	94	100	92
10 (十)	sap6	46	92	60	78	86	90

Table 4. Recognition performance of 11 Cantonese digits "0" to "10" (test data) by the 3-layer fully connected neural network with recurrent paths.

h_1		8	12	14	16	18	22
Digit	Syllable	Performance (%)					
2 (二)	ji6	98	100	100	100	100	100
10 (十)	sap6	46	92	60	34	86	90
12 (十二)	sap6, ji6	94	78	86	80	90	90
20 (二十)	ji6, sap6	94	98	96	98	88	98

Table 5. Recognition performance of the sequential Cantonese digits "2", "10", "12" and "20" (test data) by the 3-layer fully connected neural network with recurrent paths.

h_1	8	12	14	16	18	22
Testing Performance (%)	89.4	90.2	91.1	91.7	92	95.4
Training Performance (%)	99.6	98.5	98.8	98.8	99.6	98.8

Table 6. Overall recognition performance of 13 testing and training Cantonese digits by the 3-layer fully connected neural network with recurrent paths.

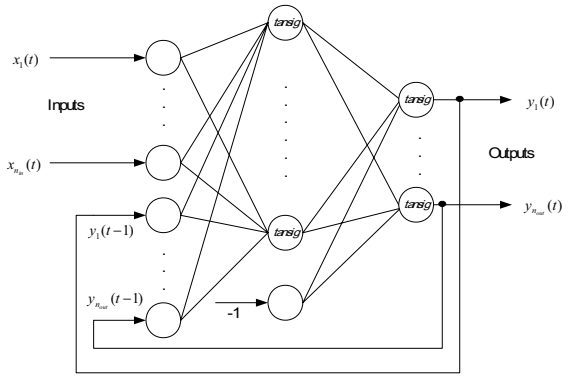


Figure 8. 3-layer fully connected neural network with recurrent paths.

6. Conclusion

A proposed Cantonese-digit speech recognizer by using a GA-based modified dynamic recurrent neural network has been developed. The structure of the modified neural network consists of two parts: a rule-base 3-layer feed-forward neural network and a classifier 3-layer recurrent neural network. The network parameters are trained by an improved GA. With this specific network structure, the dynamic feature of the speech signals can be generalized and the parameter values of the network can adapt to the values of the input data set. Cantonese digits 0 to 10, 12 and 20 have been used to demonstrate the merits of the proposed network. By using the proposed dynamic network, the dynamic and static information of the speech can be modeled effectively. Therefore, both single-syllable and multi-syllable Cantonese digits can be recognized.

7. Acknowledgement

The work described in this paper was substantially supported by The University of Western Australia, Australia, and a grant from the Centre for Multimedia Signal Processing, The Hong Kong Polytechnic University (Project No. A432).

8. References

Brown, M. & Harris, C. (1994). *Neuralfuzzy Adaptive Modeling and Control.* Prentice Hall, 1994.

Davis L. (1991). *Handbook of Genetic Algorithms.* NY: Van Nostrand Reinhold.

Engelbrecht, A.P. (2002). *Computational Intelligence: An Introduction*, John Wiley & Sons, Ltd, England.

Hanaki, Y., Hashiyama, T. & Okuma, S. (1999). "Accelerated evolutionary computation using fitness estimation," in *Proc. IEEE Int. Conf. Systems, Man, and Cybernetics*, 1999, vol. 1, pp. 643-648.

Jang, R.J.S. (1997). *Neural-Fuzzy and Soft Computing*. NJ: Prentice Hall.

Kirschning, I, Tomabechi, H., Koyama, M., & Aoe, J.I. (1996). "The time-sliced paradigm: a connectionist method for continuous speech recognition," *Information Sciences*, vol. 93, issues 1-2, pp. 133-158, Aug. 1996.

Markowitz, J.A. (1996). *Using Speech Recognition.* New Jersey, Prentice Hall.

Michalewicz, Z. (1994). *Genetic Algorithm + Data Structures = Evolution Programs*, 2nd extended ed. Springer-Verlag.

Lam, H.K., Ling, S.H., Leung, F.H.F., & Tam, P.K.S. (2004). "Function estimation using a neural-fuzzy network and an improved genetic algorithm," *International Journal of Approximate Reasoning*, vol. 30, no. 3, pp. 243-260, Jul. 2004.

Lee, T., Ching, P.C. & Chan, L.W. (1995). "Recurrent neural networks for speech modelling and speech recognition," *Acoustic, Speech and Signal Processing*, 1995 Int. Conf. on ICASSP-95, vol. 5, May 1995, pp. 3319-3322.

Lee, T., Ching, P.C. & Chan, L.W. (1998). "Isolated word recognition using modular recurrent neural networks," *Pattern Recognition*, vol. 31, no. 6, pp. 751-760, 1998.

Leung, F.H.F., Lam, H.K., Ling, S.H. & Tam, P.K.S. (2003). "Tuning of the structure and parameters of neural network using an improved genetic algorithm," *IEEE Trans. Neural Networks*, vol.14, no. 1, pp.79-88, Jan. 2003.

Leung, F.H.F., Lam, H.K., Ling, S.H. & Tam, P.K.S. (2004). "Optimal and stable fuzzy controllers for nonlinear systems using an improved genetic algorithm," *IEEE Trans. Industrial Electronics*, vol. 51, no. 1, pp.172-182, Feb. 2004.

Ling, S.H., Leung, F.H.F., Lam, H.K., & Tam, P.K.S. (2003). "Short-term electric load forecasting based on a neural fuzzy network," *IEEE Trans. Industrial Electronics*, vol. 50, no. 6, pp.1305-1316, Dec. 2003.

Pham, D.T. & Karaboga, D. (2000). *Intelligent Optimization Techniques, Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks.* Springer.

Rabiner L., & Juang, B.H. (1993). *Fundamentals of Speech Recognition*. Englewood Cliffs, New Jersey, Prentice Hall.

Setnes, M. & Roubos, H. (2000). "GA-fuzzy modeling and classification: complexity and performance**,"** *IEEE. Trans, Fuzzy Systems*, vol. 8, no. 5, pp. 509–522, Oct. 2000.

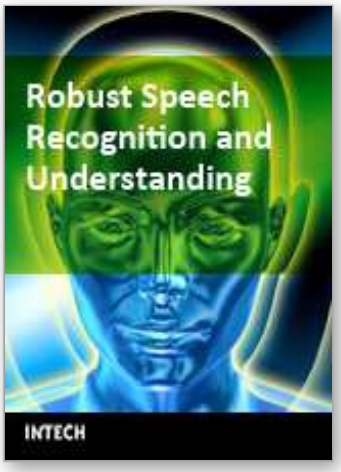
Srinivas, M. &. Patnaik, L.M. (1994). "Genetic algorithms: a survey**,**" *IEEE. Computer*, vol. 27, issue 6, pp. 17-26, June 1994.

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., & Lang, K.J. (1989). "Phoneme recognition using time-delay neural networks," *IEEE. Trans. Acoust., Speech, Signal Processing*, vol.37, pp.328–339, Mar. 1989.

Wu, J., & Chan, C. (1993). "Isolated word recognition by neural network models with cross-correlation coefficients for speech dynamic," *IEEE. Trans. on Pattern Analysis and Machines Intelligence*, vol. 15, no. 11, Nov. 1993.

Zhang, L.P., Li, L.M. & Cai, C.N. (1993). "Speech recognition using dynamic recognition neural network," in Proc. *Computer, Communication, Control and Power Engineering, 1993 IEEE. Region 10 Conf.* (TENCON '93), vol. 3, 19-21 Oct 1993, pp 333-336.

Zhang, G.P. (2000). "Neural networks for classification: A survey," *IEEE Trans. on Sys. Man. and Cyber.*, part C, vol. 30, no. 4, Nov. 2000, pp 451-462.



Robust Speech Recognition and Understanding

Edited by Michael Grimm and Kristian Kroschel

ISBN 978-3-902613-08-0
Hard cover, 460 pages
Publisher I-Tech Education and Publishing
Published online 01, June, 2007
Published in print edition June, 2007

This book on Robust Speech Recognition and Understanding brings together many different aspects of the current research on automatic speech recognition and language understanding. The first four chapters address the task of voice activity detection which is considered an important issue for all speech recognition systems. The next chapters give several extensions to state-of-the-art HMM methods. Furthermore, a number of chapters particularly address the task of robust ASR under noisy conditions. Two chapters on the automatic recognition of a speaker's emotional state highlight the importance of natural speech understanding and interpretation in voice-driven systems. The last chapters of the book address the application of conversational systems on robots, as well as the autonomous acquisition of vocalization skills.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821